

Saved: Tuesday, February 17, 1998 11:37:29 PM

Version 0.7, February 1998

Copyright © 1996-1998 Christopher E. Hyde. All rights reserved.

drjekyll@hilight.demon.co.uk

<http://www.hilight.demon.co.uk/>

Contents

Welcome

Licence

How You Can Help

About the Author

Release Notes

Installing Dialog Director

Auto Dialogs and Live Dialogs

Event Handlers

dd auto dialog - create, display and interact with a dialog window

dd calc dialog bounds - calculate the bounding rectangle of a dialog window

dd count dialogs - return the number of open dialogs

dd delete - delete dialog items or close and delete a dialog window

dd get - retrieve properties from objects

dd install- create and initialise the Dialog Director environment

dd interact with user - interact with the front dialog window

dd make - create one or more dialog items in an open dialog

dd make dialog - create and display a dialog window

dd set - set properties of objects

dd uninstall - clean up and remove the Dialog Director environment

Object Classes

dialog - data used to create a dialog

dialog item - a dialog item

push button - a push button item

check box - a check box item

radio button - a single radio button item

radio group - a group of radio buttons

pop up - a pop-up menu item

list box - a scrolling, text list

icon list box - a scrolling, labeled, icon list

text field- an optionally labelled editable text item

password field - an opaque editable text item

static text - a static text item

group box - an optionally labelled rectangular frame item

pict - a static picture item

icon - a static icon item

dummy - a dummy dialog item

color picker - a colour picking item

gauge - a graphical magnitude/progress indicator

generic control - a generic control item

icon push button - a push button with an icon & a beveled edge

icon toggle button - a toggle button with an icon & a beveled edge

icon sticky button - a sticky button with an icon & a beveled edge

icon radio button - a radio button with an icon & a beveled edge

poly push button - a clickable polygon area

poly toggle button - a toggling polygon area

poly sticky button - a sticky polygon area

poly radio button - a radio button polygon area

movie controller - a QuickTime movie controller

font spec - a text font specification

The Global Font Spec Table

Dependencies

Error Codes

Dialog Dumper

Introductory Examples

Contents of 'Examples' Folder

Known Problems and Limitations

Other Scripting Additions

Welcome

Welcome to Dialog Director v0.7. This is the February 1998 public release of the Dialog Director scrip

Please use Dialog Director and tell me what you think. I intend to enhance and maintain it based on th

The reason that this release is called v0.7 and not v1.0 is because I am still adding features and the

Current features include

Dialog item types

- Push buttons - with user defined default button, return & escape key mapping, actions

- Check boxes

- Radio button groups - one & two dimensional automatic layout

Saved: Tuesday, February 17, 1998 11:37:29 PM

- Pop-up menus
- Scrolling text lists - single column, single & multiple selection, keyboard navigable, disableable
- Static text items - any font, size, style & colour
- Edit text items- optional label, forward & backward tabbing, cut, copy & paste, extended keyboard su
- Password text items - as edit text items but display only bullets
- Rectangular frames- optional label, 3 styles (and horizontal & vertical lines)
- Pictures
- Icons - colour or black & white
- Editable pop-ups
- Colour pickers
- Gauges (progress bars & barber poles)
- Icon list boxes
- Icon buttons (push, toggle, sticky & radio)
- Polygon buttons (push, toggle, sticky & radio)
- QuickTime movie controller

Other

- Dependencies - automatically enable and disable items depending on other items' states
- Optional dialog time-out
- Extensive error checking and reporting
- Returns an easy to read list of item values
- No need for resource files or use of ResEdit
- Six different window styles including movable modal
- Floating windows - windows that float above all applications [experimental]
- Support for Apple Greyscale(ish) Appearance (AGA)
- Embedded sub-dialogs
- Separate make, interact and delete dialog events
- Get and set values & other properties of dialog items while the dialog is open
- Multiple, independent dialogs
- Get and set multiple item values & other properties via a single call
- Dynamically create and delete multiple dialog items in an active dialog
- Font spec table- supports any font spec for any item
- Make a dialog window float above a specific application

Future features

- Tristate buttons - on, off, neither
- Sliders
- Number text fields
- Balloon help
- Keyboard to item mapping
- Styled text edit
- And too many other items to list here :-)
- What would you like Dialog Director to do for you?

Licence

Please note that this software is provided "as is" and without any express or implied warranties, incl
Dialog Director is FREE. You may redistribute Dialog Director to anyone as long as you include the co
The "complete package" is defined as meaning the entire and exact contents of the file <http://www.hyld.com>
Although not a formal part of the licence I strongly suggest that if you use any URLs to Dialog Directo

How You Can Help

I want Dialog Director to be compatible with as many different machines, Mac OS versions, and language

About the Author

Hello, my name is Christopher E. Hyde. I am the creator of Dialog Director. I have over 10 years of c
Just for your information: Dialog Director contains approximately 15,000 lines of code written in C++

Release Notes

This section briefly lists the changes between the different releases of Dialog Director.

Dialog Director v0.7 (Feb '98)

- Fixed update problem with dd set in text field & password field items.
- Fixed dd set value of **<radio group>** to match dd get value.
- Fixed a few memory leaks and problems under error conditions.
- Renamed document window window style to standard window.
- Renamed modal window style to standard dialog.
- Renamed movable modal window style to movable dialog.
- Renamed palette window style to standard palette.
- Replaced default property of class dialog with default item.
- Replaced the font parameter of dd auto dialog & dd install with with fonts. This defines a list of f
- Replaced the floating parameter of dd auto dialog & dd install with float above. This allows the sp
- Improved look for greyscale pop up items.
- Improved look for indeterminate mode gauge items (barber pole).
- Improved look for labeled group box items.
- Added support for up to 32K bytes of text in text field, password field and static text items.
- Added new icon button dialog items (icon push button, icon toggle button, icon sticky button, icon r
- Added new polygon button dialog items (poly push button, poly toggle button, poly sticky button, pol
- Added new generic control dialog item.
- Added new movie controller dialog item.
- Added new icon list box dialog item.

Saved: Tuesday, February 17, 1998 11:37:29 PM

- Added new radio button dialog item.
- Improved code optimisation - its faster.
- Improved memory utilisation - lower minimum memory requirements.
- Changed dd delete event. Delete one or more dialog items from a live dialog window.
- Changed dd get event. Get a list of values of a property from a range of dialog items.
- Changed dd set event. Set the value of a property of a range of dialog items from a list.
- Changed value property. It is now compatible with script debugger.
- Changed behaviour of default item property. 0 Æ no default or cancel button, -1 Æ first push button
- Added set/get bounds property of dialog window.
- Added set selection property of text & password fields.
- Added set font property for all dialog items.
- Added set max value property of gauge items.
- Added set contents property of pop up items.
- Added new dd make dialog items event. Dynamically add one or more dialog items to a dialog window.
- Added new closeable property to class dialog.
- Added new float above option to dd install and dd auto dialog events.
- Added new action property to class list box. Return this value if list is double-clicked.
- Added new column widths property to class list box. Aligns text containing multiple tabs.
- Added support to text field for <command> + 'A' selecting all.
- Added support for named action handlers via applet's script or given script:<script>.
- Changed setting of value property of a list box to scroll first selected item into view.
- Added reporting of problem dialog item index in dd auto dialog, dd make and dd make dialog.
- Changed DD's window kind. This should make it compatible with more applications.
- Added back default button outline in floating windows including Appearance Manager support.

Dialog Director v0.6 (Apr '97)

- Fixed grey background problem in text field & password field items.
- Fixed typing return in text fields.
- Improved redraw speed by pruning unnecessary object redrawing.
- Added <command> + '.' to cancel button mapping.
- Removed colour synonym for color (due to reports of strange behaviour in QuarkExpress).
- Renamed do dialog event to dd auto dialog.
- Renamed calc window bounds event to dd calc dialog bounds.
- Added new dd install event. Create and initialise the Dialog Director environment.
- Added new dd make dialog event. Create and display a dialog window.
- Added new dd interact with user event. Interact with the front dialog window.
- Added new dd get event. Get a property from a dialog item.
- Added new dd set event. Set a property of a dialog item.
- Added new dd delete event. Close and delete a dialog window.
- Added new dd uninstall event. Clean up and remove Dialog Director environment.
- Added new dd count dialogs event. Return the number of open dialogs.
- Changed the items property of dialog to contents.
- Changed label property of push button to name.
- Changed label property of check box to name.
- Changed the items property of radio group to contents.
- Changed label property of pop up to name.
- Completely rewrote pop up class. No longer uses System pop-up control.
- Added support to pop up for optional type-in text field.
- Added support to pop up for Apple Greyscale(ish) Appearance (AGA).
- Changed label width property of pop up to name width.
- Changed the items property of pop up to contents.
- Added support to list box for initial multiple selections.
- Changed the items property of list box to contents.
- Changed label property of text field to name.
- Changed label bounds property of text field to name bounds.
- Changed label property of password field to name.
- Changed label bounds property of password field to name bounds.
- Changed label property of static text to contents.
- Changed label property of group box to name.
- Changed data property of pict to contents.
- Changed data property of icon to contents.
- Added support to radio group for dependencies.
- Changed depends on property of all dialog items to enabled.
- Added support to for initially disabled items.
- Added new color picker class.
- Added new gauge class.

Note: The 'label' property was replaced with 'name' (except in the static text class where it was repl

Dialog Director v0.5.1 (6 Dec '96)

- Fixed tabbing direction. (Tab & Shift-Tab were transposed in v0.5 :-).
- Fixed documentation and example inconsistencies. Specifically condense Æ condensed and centered Æ c
- Removed default button outline in floating windows.
- Changed item drawing order to be the same as the item list order.
- Fixed minor sub-window update-on-close problem.
- Added full support for pict and icon data properties.

Saved: Tuesday, February 17, 1998 11:37:29 PM

- Added support for primary, secondary & tertiary style separator lines.
- Added standard notification when in background and trying to opening a window (actually in 0.5 but f
- Changed grey/colour drawing to take place on > 2 bit deep displays only. (Previously some were > 2,
- Added support for typing carriage returns in text multi-line fields.
- Added 'last item of the result' is the final bounds rectangle of the window.
- Added calc window bounds event handler, to calculate the initial window bounds.
- Added support for dialog items with dependencies on specific list box items.
- Fixed an annoying TextEdit 'feature' if " or & is pressed when a selection exists.
- Improved the dictionary (I hope :-).
- Added some more examples.
- Included Resource Utilities v1.0b1 osax and examples (no documentation yet).

Dialog Director v0.5 (25 Nov '96)

This release includes a few fixes, many new features, some changes and a couple of experimental items

- Changed all property and class 4 character codes (in an attempt to avoid future terminology conflict
- Added action property to push button class. An action may be a dialog record or a script handler [ex
- Added font spec class with name, size, style and color properties.
- Added floating parameter to do dialog [experimental]. This makes all your DD windows float above al
- Added font parameter to do dialog.
- Added greyscale parameter to do dialog, support for Apple Greyscale Appearance (AGA).
- Added style, name and font properties to dialog class.
- Changed pop up class to return a string when the items is a resource type such as 'FONT'.
- Changed the nothing's that were returned by do dialog to null's. (I previously did not know that a
- Replaced text font, text size & text style properties with font property in static text class.
- Added justification property to static text class.
- Replaced frame class with group box class.
- Added style property (primary group, secondary group & tertiary group) to group box class.
- Added support for enabling and disabling list boxes.
- Resolved failure of Event Manager to deliver null events when update events are pending.
- Increased maximum number of pop up menus from 10 to 16. This can now be increased even further by th
- Does not open window until application is in front (blinks Application Menu icon), or if user intera
- Addressed all (four) reported problems. There were two terminology conflicts (I suppose that I shou

Dialog Director v0.4 (27 Oct '96)

This was the first public release of Dialog Director.

Installing Dialog Director

To install Dialog Director all you have to do is copy the "Dialog Director" scripting addition file in

If you intend to make use of DD's floating windows then you may also want to install the "TSM Fix 1.03"

Now you are ready to run the sample scripts and write your own.

Auto Dialogs and Live Dialogs

Dialog Director supports two different ways of managing dialog windows. These have been named "Auto D

Auto Dialogs

Automatic (or autonomous) dialogs are those where a single call to dd auto dialog... creates & displays

Live Dialogs

Live dialogs were new in DD v0.6 and require separate calls:

- to create and initialise the Dialog Director environment (dd install...),
- to create & display the dialog (dd make dialog...),
- to repeatedly interact with the user (dd interact with user...),
- to get properties from dialog items (dd get...),
- to set properties of dialog items (dd set...),
- to close and delete the dialog (dd delete dialog...),
- and to finally clean up and remove the Dialog Director environment (dd uninstall).

Live dialogs allow the management of moderately complex to very advanced dialogs where many user action

The essential, live dialog, AppleScript loop is:

```
dd install -- Suffix with any options
```

```
set d to dd make dialog <a dialog record>
```

```
repeat
```

```
  set i to dd interact with user -- Wait for user input
```

```
  if i = <index of an exit item> then
```

```
    exit repeat
```

```
  else if i = <index of an item that requires script support> then
```

```
    -- Update necessary dialog items, open another dialog, etc. E.g.:
```

```
    dd set <a property> of item <an index> of d to <a value>
```

```
  end if
```

```
end repeat
```

```
set theResult to dd get value of every item of d -- Get all dialog item values
```

```
dd delete d -- Remove the dialog
```

```
dd uninstall -- Sometime before we quit
```

Event Handlers

This section describes the eleven event handlers contained in the Dialog Director suite. It includes

Notes and Documentation Conventions

- All parameters marked with an asterisk* are optional.
 - The default value of a parameter is shown in parenthesis following the description.
 - Typing <command> + <control> + 'Q' will safely terminate a dd auto dialog or dd interact with user c
- dd auto dialog: Create, display and interact with a dialog window

Saved: Tuesday, February 17, 1998 11:37:29 PM

```
dd auto dialog      record The dialog description record.  (see class dialog)
  with fonts*      list of font specGlobal font table.  (see class font spec)
  grayscale*       boolean   Use Apple greyscale(ish) appearance.  Greyscale is a synonym of grayscale.
  float above*     application Make windows float above all others of one or all applications.  (not f
  given script:*   script Use this script to resolve action handler names.  (no script)
Result:           list of anything The final dialog item values and bounds of the dialog window.
```

Example:

```
set aDialog to {size:[320, 95], timeout after:60, contents:[-
  {class:push button, bounds:[250, 65, 310, 85], name:"OK"}, -
  {class:push button, bounds:[170, 65, 230, 85], name:"Cancel"}, -
  {class:static text, bounds:[10, 10, 310, 10 + 32], contents:"A very simple dialog box.
Just testing!"]}]}
```

```
get dd auto dialog aDialog with grayscale
```

This event creates an Auto Dialog containing the items described in the dialog record, it displays the dialog. The with fonts parameter specifies a list of font spec records that defines the global font table. The Setting the grayscale parameter to true forces all DD windows to be drawn with grey backgrounds and it makes the dialog float above all other windows. The float above parameter makes the window float above either a single application (e.g. float above a specific application) or all applications. The given script: parameter is experimental. It is used to define a parent script context that contains the dialog. Note: All the optional parameters of a dd auto dialog call are ignored if it is made between calls to dd calc dialog bounds. Warning: There is a known problem when using Script Debugger v1.0.x. If, from a script run in Script Debugger, you call dd calc dialog bounds: Calculate the bounding rectangle of a dialog window

```
dd calc dialog bounds
  point Size of dialog: [width, height].
Result:  rectangle The calculated rectangle.
Example:
set dRect to dd calc window bounds [260, 95]
```

Use this event to pre-calculate the bounds property of a dialog. The rectangle returned will depend on the dialog's size and position. dd count dialogs: Return the number of open dialogs

```
dd count dialogs
Result:  integer The number of open dialogs.
```

This event returns the current number of open DD windows in the target application. If DD is not currently running, it returns 0. dd delete: Delete dialog items or close and delete a dialog window

```
dd delete reference The dialog or dialog item(s) to delete.
```

Examples:

```
set dlog to dd make dialog ... -- dlog is now a dialog ref of the form 'dialog id #'

-- Deleting dialog Items
dd delete item 3 of dialog 1 -- Delete the 3rd dialog item of the front most dialog

dd delete first item of dlog -- Delete the first dialog item of dialog 'dlog'

dd delete items 2 thru 4 of dlog -- Delete dialog items 2, 3 & 4 of dialog 'dlog'

dd delete items -3 thru -1 of dialog id 1-- Delete the last 3 dialog items of the back most dialog 'dlog'
```

```
-- Deleting dialogs
dd delete dialog 1 -- Delete the front most dialog
- or -
dd delete dlog -- Delete the dialog 'dlog' and any in front of it
- or -
dd delete dialog id 1 -- Delete all the dialogs (as dialog id 1 is the back most dialog)
```

This event has two separate functions:

1. Dialog Items: It dynamically deletes one or more dialog items from an active dialog. Once deleted, the items are no longer available.
2. Dialogs: It closes and deletes the specified dialog and any other dialogs in front of it and all the dialogs behind it.

This event may only be called between dd install and dd uninstall.

```
dd get: Retrieve properties from objects
dd get reference The property to be returned and the object(s) from which to retrieve it.
Result:  anything The data from the dialog item.
```

Examples:

```
set dlog to dd make dialog ... -- dlog is now a dialog reference
```

```
set n to dd get value of item 7 of dlog -- Retrieve the value of dialog item 7 of dlog
set n to dd get value of dialog item 7 of dialog 1 -- Same as above
set dVals to dd get value of every item of dlog -- Retrieve the value of every item of dlog and the dialog
set someVals to dd get value of items 3 thru -1 of dialog 1 -- Retrieve the value of items 3 to (conten
```

```
set windowRect to dd get bounds of dlog -- Get the current bounds of the dialog window
```

Saved: Tuesday, February 17, 1998 11:37:29 PM

This event allows the reading of properties of dialog items in any currently open dialog. It retrieves a list containing the current value properties of any contiguous range of dialog items in any open dialog. See `dd make dialog` below for details of dialog numbers and IDs. This event may only be called between `dd make dialog` and `dd delete d`.
 Note: Currently the only valid dialog item property to get is value.

Note: The terms item and dialog item are freely interchangeable when referring to items in a DD window.

`dd install`: Create and initialise the Dialog Director environment

`dd install`

```
with fonts*    list of font specGlobal font table. (see class font spec)
grayscale*     boolean    Use Apple greyscale(ish) appearance. Greyscale is a synonym of grayscale
float above*   application Make windows float above all others of one or all applications. (not f
given script:*  script Use this script to resolve action handler names. (no script)
```

Example:

`dd install with grayscale`

The `dd install` event creates and sets up the DD environment within the currently targeted application.

The `dd install` call is necessary prior to any calls to `dd make dialog`, `dd interact with user`, `dd get`, or `dd delete d`.

The `with fonts` parameter specifies a list of font spec records that defines the global font table. The list is a list of font spec records.

Setting the `grayscale` parameter to true forces all DD windows to be drawn with grey backgrounds and it forces all dialog items to be drawn with grey backgrounds.

The `float above` parameter makes all DD windows float above either a single application (e.g. float above application) or all applications.

The `given script:` parameter is experimental. It is used to define a parent script context that contains the script to be used to resolve action handler names.

Note: All the optional parameters of a `dd auto dialog` call are ignored if it is made between calls to `dd make dialog` and `dd delete d`.

Warning: There is a known problem when using Script Debugger v1.0.x. If, from a script run in Script Debugger, you call `dd interact with user`, the script will hang.

`dd interact with user`: Interact with the front dialog window

`dd interact with user`

```
for max ticks*  integer    The maximum ticks before returning null if the user has not interacted.
```

Result: anything The index of the dialog item clicked or null.

Example:

`set d to dd make dialog ...`

`repeat`

```
set i to dd interact with user -- Wait for user input
```

```
if i = 1 then -- Cancel button
```

```
    exit repeat
```

```
else if i = 2 then -- Start button
```

```
    StartSomething()
```

```
else if i = 3 then -- Stop button
```

```
    StopSomething()
```

```
else
```

```
    -- Handle other items here
```

```
end if
```

`end repeat`

`dd delete d -- Remove the dialog`

This event enables and handles any user interaction with the front most Live Dialog. Interaction consists of clicking on a dialog item.

If the front dialog window has a close box (in the top left corner of its window frame) then `dd interact with user` will return 1.

`dd make`: Create one or more dialog items in an open dialog

`dd make` dialog item(s) A record or list of records that describe the dialog items.

```
at location reference    The dialog into which to insert the items.
```

Example:

```
-- Create a push button in the back most window
```

`dd make {class:push button, bounds:[20, 65, 80, 85], name:"One"} at dialog -1`

This event allows the dynamic creation of one or more dialog items in an open dialog. Currently the only valid dialog item property to set is value.

`dd make dialog`: Create and display a dialog window

`dd make dialog` record A record that describes the dialog.

Result: dialog reference A reference to the new dialog.

Example:

`set dA to dd make dialog ... -- dA is dialog id 1 = dialog 1`

`set dB to dd make dialog ... -- dB is dialog id 1 = dialog 2`

```
-- dB is dialog id 2 = dialog 1
```

`set dC to dd make dialog ... -- dA is dialog id 1 = dialog 3`

```
-- dB is dialog id 2 = dialog 2
```

```
-- dC is dialog id 3 = dialog 1
```

This event creates and displays a Live Dialog containing the items described in the dialog record. A reference to the dialog is returned.

The reference returned by `dd make dialog` is valid until the dialog is deleted. This event handler may be used to delete the dialog.

`dd set`: Set properties of objects

`dd set reference` The property and object(s) to be changed.

```
to anything    The new property value.
```

Examples:

`dd set value of item 3 of dialog 1 to "Some Text"`

Saved: Tuesday, February 17, 1998 11:37:29 PM

```
dd set value of items 2 thru 6 of dialog 1 to ["A String", 1, null, myName, false]

set theListItem to a reference to item 5 of dialog id 3 -- A list box item
dd set contents of theListItem to theNewListContents
dd set value of theListItem to theNewListContents's length -- Select last item

dd set bounds of theMovingButton to [x, y, x + 60, y + 20]
dd set name of theOKButton to "Not OK"
dd set enabled of theOKButton to (true) -- Parenthesis to stop AppleScript changing it
dd set contents of theStaticText to "Not so static, Huh!"

dd set bounds of dialog 1 to theNewBounds
```

This event allows the setting of many different properties of dialog items in any currently open dialog. A dialog item's index is its position in the dialog's contents property. The dialog item indices used. The same property of any contiguous range of dialog items in any open dialog may be set by using a single dd set. See dd make dialog above for details of dialog numbers and IDs. This event may only be called between dd install and dd uninstall: Clean up and remove the Dialog Director environment

dd uninstall

Example:

dd uninstall

This event cleans up and removes the DD environment created by an earlier call to dd install. The cleanup. Tip: If, during script development, you encounter a script error and the script terminates leaving an incomplete dialog, dd uninstall will clean up the environment.

Object Classes

This section describes the many object classes contained within the Dialog Director suite. A complete description of the Dialog Director class hierarchy

Dialog Director class hierarchy

Notes and Documentation Conventions

- All properties marked with an asterisk* are optional.
- The default value of a property is shown in parenthesis following the description.
- All of the help properties are currently ignored.
- All text/string properties are currently limited to a maximum of 255 bytes (characters?) in length.
- The 'label' property was previously called 'name' but I cannot decide whether it should be called 'name' or 'label'.
- Properties marked with [W] are writeable while a dialog is open (via dd set).
- Properties marked with [R/W] are readable and writeable while a dialog is open (via dd get and dd set).
- Rectangles are always specified [left, top, right, bottom] relative to the dialog window's origin (via dd set).

Class dialog: data used to create a dialog

Elements:

dialog item by numeric index

Properties:

size* point Size of window: [width, height]. Specifying a window size forces the dialog box to be a certain size.

bounds* rectangle Bounds of window. (rect of main screen inset by 16 pixels) [R/W]

style* window frame style The kind window and style of window frame. One of: StyleDescription, StyleDescription, StyleDescription, StyleDescription

closeable* boolean Indicates whether the window has a close box. This only functions for dialog windows.

name* string The title of window. This is displayed in some of the styles of window frame. ("")

help* string Help string. [currently unused]

timeout after* integer Seconds after which the window closes automatically. (no timeout)

font* font spec [Obsolete]

default item* integer Index of the default (outlined) push button. (first push button item)

contents a list of dialog item The dialog items contained in this window.

Example:

```
dd auto dialog {size:[260, 95], timeout after:60, default item:3, contents:dItems}
```

The size and bounds properties are mutually exclusive. The bounds property of a dialog may now be readable and writeable.

Note that the style of the window has no bearing on its modal/modeless nature. All dialog director windows are modal.

If the optional closeable property is set to true then clicking in the window's close box will either close the window or make it modal.

The optional default item property specifies the index of the default push button item. Setting it to 0 means no default button.

The contents property is a list of records that specifies the initial dialog items that appear in the dialog. Each record is a list of 4 items:

Class dialog item: a dialog item

Plural form:

dialog items

Properties:

class class This property must be set to the class of the required item.

bounds rectangle The bounding rectangle of the item: [left, top, right, bottom]

font* integer The index of the font spec to use for this item's text. (varies with class) [W]

help* string The help text. ("") [currently unused]

enabled* boolean, integer or list True, false, or the other items on which this item depends.

All sub-classes of dialog item inherit the properties of this class (with the exception of dummy item class).

Dialog items that display text will display that text using the font spec specified via the font property.

The depends on property (from v0.5.1) has been replaced with the enabled property. The enabled property is a list of other dialog items on which this item depends.

Class push button: a push button item

Properties:

Saved: Tuesday, February 17, 1998 11:37:29 PM

```

class class This property must be set to push button.
bounds rectangle The bounding rectangle of this button: [left, top, right, bottom] [W]
font* integer The index of the font spec to use for the button's name. (2) [W]
help* string The help text for this button. (") [currently unused]
enabled* boolean, integer or list True, false, or the other items on which this button depends.
name string The title/name of this button. [W]
action* anything Button's action when pressed. (no action)

```

Return value:

If no action is specified:

boolean True if this button was used to dismiss the dialog, otherwise false.

If an action specified:

anything The result of the action if the button was pressed, otherwise null.

Example:

```
set pb1 to {class:push button, bounds:[430, 390, 490, 410], name:"OK", help:"Dismiss"}
```

Currently all push button items dismiss the dialog. If no default button is specified in the dialog r

The action property defaults to null meaning that there is no action. For auto dialog this also indic

- a dialog record - in which case a new dialog is opened modally on top of the current one. When dis
- a handler script/function - the handler (which must have no parameters) is executed and the result

```
on GetFile()
```

choose file with prompt "Select a file:"

```
end GetFile
```

```
set pb3 to {class:push button, bounds:[100, 50, 160, 70], name:"Open...", action:GetFile}
```

- a string that is a handler's name - the handler which must be in the parent script context (see sec

```
property gPresses : 0
```

```
on DoButton given item:i
```

set gPresses to gPresses + 1

display dialog "DoButton: Item " & i & " was pressed. Presses = " & gPresses

```
end DoButton
```

```
on AnAction()
```

display dialog "AnAction: A push button was pressed."

```
end AnAction
```

```

dd auto dialog {size:[80, 130], contents:[{class:push button, bounds:[10, 100, 70, 120], name:"OK"},
{class:push button, bounds:[10, 10, 70, 30], name:"One", action:"dobutton"}, ↵
{class:push button, bounds:[10, 40, 70, 60], name:"Two", action:"dobutton"}, ↵
{class:push button, bounds:[10, 70, 70, 90], name:"Three", action:"anaction"}]} ↵
given script:me -- Use this only when running from a script editor. Remove before saving as an ap

```

Any errors that occur in an action are propagated through and returned by the dd auto dialog/dd intera

Tip: To set a push button to do a one shot action make it inversely dependent upon itself, and ensure

```
on Bang()
```

return 1

```
end Bang
```

```
{class:push button, bounds:[0, 0, 60, 20], name:"One Shot", action:Bang, enabled:-5}
```

The script may also be written:

```
on Bang()
```

dd set enabled of item 5 of dialog 1 to (false)

```
end Bang
```

```
{class:push button, bounds:[0, 0, 60, 20], name:"One Shot", action:Bang}
```

Note: The value property of a push button may be set only indirectly via an action handler's return st

Class check box: a check box item

Properties:

class class This property must be set to check box.

bounds rectangle [W]

font* integer The index of the font spec to use for the button's name. (2) [W]

help* string

enabled* boolean, integer or list [W]

name string [W]

value* boolean If true then the check box is checked. (false) [R/W]

Return value:

Saved: Tuesday, February 17, 1998 11:37:29 PM

```
set dTypeInPopUp to {size:[120, 80], contents:[~
  {class:push button, bounds:[50, 50, 110, 70], name:"OK"}, ~
  {class:pop up, bounds:[95, 10, 130, 29], contents:"9;10;12;14;18;24;36;48", text field:3}, ~
  {class:text field, bounds:[50, 12, 90, 28], name:"Size:", name bounds:[10, 12, 50, 28], value:9}]}
dd auto dialog dTypeInPopUp with grayscale
```

Currently each pop-up menu requires a dummy menu resource (type 'MENU'). There are 16 menu resources. The Pop up dialog item provides a pop-up menu style control. DD v0.6 completely reimplements pop-up controls. If a resource-names style menu is created (such as with contents:"FONT") then the value may be initially set to the resource name.

Class list box: a scrolling, text list

Properties:

```
class class This property must be set to list box.
bounds rectangle
font* integer The index of the font spec to use for the list's contents. (4) [W]
enabled* boolean, integer or list [W]
value* integer or list The selected item or items. (0 = no selection) [R/W]
flags* integer Selection flags for this list. (130) This property controls the selection algorithm.
```

Constant Description

```
2 disable highlighting of empty cells
4 allow use of shift key to deselect items
8 shift-drag selects items passed by cursor
16 reset list before responding to shift-click
32 prevent discontinuous selections
64 enable multiple item selection without shift key
128 allow only one item to be selected at once
```

The values of the flags property for standard list behaviours are 130 for single item selection, and 0 for multiple. The contents property is a list of strings. The list items. Each string represents a single list item. Items that are highlighted are indicated by a space before the string. The action property is an integer. The value returned when a list item is double clicked. (0 = no double click). The column widths property is a list of integers. The widths of the text alignment columns. (single column of list items).

Return value:

If only single selections are allowed:

```
integer Index of the currently selected item or 0 if no selection.
```

If multiple selections are allowed:

```
list of integers List of indices of the currently selected items or 0 if no selection.
```

List boxes may be controlled via the keyboard using the following keys (i.e. they may accept the keyboard shortcuts).

home select first item

endselect last item

up arrow move selection to previous item

down arrow move selection to next item

page up move selection up by list box height

page down move selection down by list box height

tab move keyboard focus to next focusable item

shift-tab move keyboard focus to previous focusable item

backspace delete last character in internal buffer (see other)

clear clear internal buffer (see other)

other characters typed within 1 second intervals are appended to an internal buffer which is used to construct the list items.

Example:

```
-- List the items in the System folder and force the user to select one.
```

```
set lbItems to list folder (path to system folder)
```

```
set sysFolderDlg to {size:[220, 220], contents:[~
  {class:list box, bounds:[10, 10, 210, 172], contents:lbItems, action:2}, ~
  {class:push button, bounds:[150, 190, 210, 210], name:"OK", enabled:1}]}
dd auto dialog sysFolderDlg
```

Setting the action property to a integer other than 0 indicates that when a list item is double-clicked, the column widths property allows the creation of a list box that uses tab characters in the list's contents. The height of a list box should be a multiple of the row height + 2 (for the frame). For the standard list box, the height is 172.

Note: The value property may now be initialised to a list of cell indices to highlight.

Note: Setting the value property scrolls the list to the nearest highlighted item.

Class icon list box: a scrolling, optionally labelled, icon list

Properties:

```
class class This property must be set to icon list box.
bounds rectangle
font* integer The index of the font spec to use for the list's contents' labels. (4) [W]
enabled* boolean, integer or list [W]
value* integer or list The selected item or items. (0 = no selection) [R/W]
flags* integer Selection flags for this list (see class list box above). (130)
contents list of integers The list of icon family IDs constituting the list's items. The labels are the list of strings.
action* integer The value returned when a list item is double clicked. (0 = no double click)
```

Return value:

If only single selections are allowed:

```
integer Index of the currently selected item or 0 if no selection.
```

Saved: Tuesday, February 17, 1998 11:37:29 PM

```

if multiple selections are allowed:
    list of integers List of indices of the currently selected items or 0 if no selection.
Example:
set iconIDs to [200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212]
set rf to res open alias "A Disc:A Folder:An Icon Resource File" with keep in chain
-- The icon resources IDs 200-212 come from the above file
dd auto dialog {size:[200, 312], contents:{class:icon list box, contents:iconIDs, action:1, bounds:[1
res close rf

```

The icon list box class provides a vertical, single column, scrolling list of centred and labelled icons. The height of an icon list box should be a multiple of the row height + 2 (for the frame). The row height is the height of the text field. Class text field: an optionally labelled editable text item

Properties:

```

class class This property must be set to text field.
bounds rectangle
font* integer The index of the font spec to use for the button's name. (4) [W]
enabled* boolean, integer or list [W]
name* string Label/prompt text of the item. (no name)
name bounds* rectangle Bounding box of the name text. This must be specified if a name is specified.
value* string Initial editable text of the item. (") [R/W]

```

Return value:

```
string The current editable text of the item.
```

Example:

```

set ed1 to {class:text field, bounds:[60, 160, 200, 176], name bounds:[10, 160, 60, 180], name:"Text:", value:"default text"}

```

A text field dialog item presents a standard editable text area with an optional label. Text fields and password fields fully support the extended keyboards including:

```

f2 cut
f3 copy
f4 paste
del delete forward
home move insertion point to start of text
endmove insertion point to end of text
page up move insertion point up by height of field
page down move insertion point down by height of field
clear delete selection
tabmove keyboard focus to next focusable item
shift-tab move keyboard focus to previous focusable item
command-A Select all

```

Note: Part or all of a text field or password field in an open dialog may be selected by setting its selection bounds.

```

e.g. [0,0] - put insertion point at start of field
e.g. [0,1] - select first character (from before first to after first)
e.g. [1,3] - select second & third characters (from before second to after third)
e.g. [3,-1] - select fourth character through the end of the text (from before fourth to end)
Class password field: an opaque editable text item

```

Properties:

```

class class This property must be set to password field.
bounds rectangle
font* integer The index of the font spec to use for the button's name. (4) [W]
enabled* boolean, integer or list [W]
name* string
name bounds* rectangle
value* string As for an exit text item, but always displayed as bullets. [R/W]

```

Return value:

```
string The current editable text of the item (the actual text, not the bullets).
```

Example:

```

set pwd to {class:password field, bounds:[10, 36, 250, 36 + 16], name bounds:[10, 10, 250, 26], name:"Enter Password:"}

```

A password field dialog item behaves identically to a text field dialog item except that it displays the text as bullets. Class static text: a static text item

Properties:

```

class class This property must be set to static text.
bounds rectangle
font* integer The index of the font specification to use for the contents. (3) [W]
enabled* boolean, integer or list If disabled the text appears greyish. [W]
contents string The displayed text of the item. [W]
justification* text alignment How to align/justify the text. One of: flushdefault Aligned at left Aligned at right

```

Return value:

```
null This class of item returns nothing.
```

Example:

```

dd install with fonts [null, null, null, null, {style:[bold, condensed]}, {size:18, name:"Times", style:[italic]}]

```

Saved: Tuesday, February 17, 1998 11:37:29 PM

```
-- Display "Some Text" using 18pt Times in bold, italic style
{class:static text, contents:"Some Text", bounds:[10, 10, 220, 30], font:6}
-- Display aString using the system font and size in bold, condensed style
set aString to "A sample string!"
set st1 to {class:static text, contents:aString, bounds:[410, 10, 520, 42], font:5}
```

A static text dialog item is a passive item that can display up to 32,000 characters of text (if you have a large screen).
Class group box: an optionally labelled rectangular frame or separator line item

Properties:

```
class class This property must be set to group box.
bounds rectangle
font* integer The index of the font specification to use for the contents. (3)
enabled* boolean, integer or list [W]
name* string Label/title text of the frame. (no name)
style* frame style The style in which the group box is to be drawn. (primary group) framestyle
```

Return value:

```
null This class of item returns nothing.
```

Example:

```
set box1 to {class:group box, name:"Box", bounds:[300, 16, 490, 112], style:secondary group}
```

If the width of the bounds rectangle ≤ 4 then the item is drawn as a vertical separator. If the height of the bounds rectangle ≤ 4 then the item is drawn as a horizontal separator.
In DD v0.7 labelled group boxes are drawn differently to previous releases. The bounds property specifies the bounds of the group box.
Class pict: a picture item

Properties:

```
class class This property must be set to pict.
bounds rectangle The picture is scaled to fit this rectangle.
contents picture or integer A Quickdraw picture or ID number of a picture ('PICT') resource [W]
```

Return value:

```
null This class of item returns nothing.
```

Example:

```
set pic1 to {class:pict, bounds:[10, 90, 10 + 198, 90 + 47], contents:131}
```

The res id and data properties have been replaced with the contents property. A pict dialog item draws a picture.
Class icon: an icon item

Properties:

```
class class This property must be set to icon.
bounds rectangle The icon is scaled to fit this rectangle.
contents icon or integer A colour or black & white icon or ID number of the icon ('cicn' or 'ICON')
```

Return value:

```
null This class of item returns nothing.
```

Example:

```
set icn1 to {class:icon, bounds:[160, 10, 192, 42], contents:128}
```

The res id and data properties have been replaced with the contents property. An icon dialog item draws an icon.
Class dummy: a dummy dialog item

Properties:

```
class class This property must be set to dummy.
enabled* boolean, integer or list [W]
```

Return value:

```
boolean The current state of the item, as determined by its enabled property.
```

Example:

```
-- If items 1, 2 & 3 are text or password fields then this item's state would be on
-- only when all 3 fields contained some text
```

```
set dum1 to {class:dummy, enabled:[dAnd, 1, 2, 3]}
```

The only purpose of dummy dialog items is to enable the combination of dependencies into even more complex dependencies.
Class color picker: a colour picking item

Properties:

```
class class This property must be set to color picker.
bounds rectangle
enabled* boolean, integer or list [W]
value RGB colour The colour of the picker's swatch. [R/W]
flags* integer The color picker flags. (0) [Ignored]
```

Return value:

```
RGB colour The current colour of the item's swatch.
```

Example:

```
dd auto dialog {size:[260, 100], contents:[-
{class:push button, bounds:[190, 70, 250, 90], name:"OK"}, -
{class:color picker, bounds:[10, 10, 90, 90], value:[11100, 22200, 33300]} -
]}
```

A color picker is an interactive dialog item that displays a framed, rectangular swatch of colour that can be clicked to change the colour.
The value property is in [red, green, blue] format, where 0 is no colour and 65535 is maximum colour.

Saved: Tuesday, February 17, 1998 11:37:29 PM

Class gauge: a graphical magnitude indicator

Properties:

```
class class This property must be set to gauge.
bounds rectangle
enabled* boolean, integer or list [W]
value* integer The size of the gauge indicator bar. (0) [R/W]
max value* integer The maximum value of the gauge. (100) [W]
```

Return value:

```
integer The current value of the gauge.
```

Example:

```
property theMax : 1234
dd install with grayscale
set p to dd make dialog {size:[300, 50], contents:[-
  {class:static text, contents:"Thinking...", bounds:[8, 4, 160, 20]}, -
  {class:gauge, bounds:[10, 25, 290, 25 + 12], value:0, max value:theMax} -
  ]}
repeat with n from 1 to theMax
  dd set value of item 2 of p to n
end repeat
dd delete p
dd uninstall
```

A gauge is a passive dialog item that displays an integer value as a horizontal indicator bar that fills. When max value is a positive integer then setting value to an integer between 0 and max value displays the value.

Class icon push button: a push button with an icon & a bevelled edge

Properties:

```
class class This property must be set to icon push button.
bounds rectangle
enabled* boolean, integer or list [W]
style* integer The button appearance style. (12)
contents integer An ID number of a family of icon resources. [R/W]
```

Return value:

```
boolean True if this button was used to dismiss the dialog, otherwise false.
```

Example:

```
-- A 1 pixel deep button with a small icon
{class:icon push button, bounds:[357, 274, 377, 293], contents:1001, style:5}
```

An icon push button is a dialog item that behaves similarly to a push button but displays an icon instead of text.

The depth of the icon drawn is based on the depth of the VDUs the icon intersects and the depths of the buttons. Only the required size(s) of icon need exist for a given icon family. However, even if not used directly, the icon must be defined. Setting the contents property to -1 causes the button to be drawn blank (with no icon).

Note: You can copy the icon from any "Get Info" window in the Finder and paste it into a file open in ResEdit. The style property dictates the visual construction of the button. It controls both the 3D height of the button and the depth of the icon.

or by using: (one of: large=0, small=1, mini=2) + (4 x depth)

Icon buttons should be large enough to contain the size of the specified icon and 3D border. The standard icon size is 16x16 pixels.

Class icon toggle button: a toggle button with an icon & a bevelled edge

Properties:

```
class class This property must be set to icon toggle button.
bounds rectangle
enabled* boolean, integer or list [W]
value* boolean If true then the button is on. (false) [R/W]
style* integer The button appearance style. (12)
contents integer An ID number of an icon family ('ICN#') resource. [R/W]
```

Return value:

```
boolean True if this button is in the on state, otherwise false.
```

Example:

```
-- A 1 pixel deep button with a small icon
{class:icon push button, bounds:[357, 274, 377, 293], contents:1001, style:5}
```

An icon toggle button is visually similar to an icon push button, but behaves like a check box dialog item.

Class icon sticky button: a sticky button with an icon & a bevelled edge

Properties:

```
class class This property must be set to icon sticky button.
bounds rectangle
enabled* boolean, integer or list [W]
value* boolean If true then the button is on, and is drawn highlighted. (false) [R/W]
style* integer The button appearance style. (12)
contents integer An ID number of an icon family ('ICN#') resource. [R/W]
```

Return value:

```
boolean True if this button is in the on state, otherwise false.
```

Example:

Saved: Tuesday, February 17, 1998 11:37:29 PM

```
-- A 1 pixel deep button with a small icon
{class:icon sticky button, bounds:[357, 274, 377, 293], contents:1001, style:5}
```

An icon sticky button is visually similar to an icon push button, but behaves like a single radio butt
Class icon radio button: a radio button with an icon & a bevelled edge

Properties:

```
class class This property must be set to icon radio button.
bounds rectangle
enabled* boolean, integer or list [W]
value* boolean If true then the button is on, and is drawn highlighted. (false) [R/W]
style* integer The button appearance style. (12)
contents integer An ID number of an icon family ('ICN#') resource. [R/W]
```

Return value:

```
boolean True if this button is in the on state, otherwise false.
```

Example:

```
-- A 1 pixel deep button with a small icon
{class:icon radio button, bounds:[357, 274, 377, 293], contents:1001, style:5}
```

An icon radio button is visually similar to an icon push button, but behaves like a radio button dialo
Class poly push button: a clickable polygon area

Properties:

```
class class This property must be set to poly push button.
bounds rectangle
enabled* boolean, integer or list [W]
contents* list of integer The coordinates of the polygon's vertices. (the bounding rectangle)
```

Return value:

```
boolean True if this button was used to dismiss the dialog, otherwise false.
```

Example:

```
-- A triangular push button
{class:poly push button, bounds:[195, 30, 340, 150], contents:[0, 0, 145, 60, 60, 120, 0, 0]}
```

A poly push button is a transparent polygonal area that behaves like a push button. It may be used to
The contents property defines the coordinates of the vertices of the polygon. It consists of a list o

Class poly toggle button: a toggling polygon area

Properties:

```
class class This property must be set to poly toggle button.
bounds rectangle
enabled* boolean, integer or list [W]
value* boolean If true then the button is drawn highlighted. (false) [R/W]
contents* list of integer The coordinates of the vertices.
```

Return value:

```
boolean The current state of the button.
```

Example:

```
-- A triangular toggle button
{class:poly toggle button, bounds:[195, 30, 340, 150], contents:[0, 0, 145, 60, 60, 120, 0, 0]}
```

A poly toggle button is a transparent polygonal area, similar to a poly push button, that behaves like

Class poly sticky button: a sticky polygon area

Properties:

```
class class This property must be set to poly sticky button.
bounds rectangle
enabled* boolean, integer or list [W]
value* boolean If true then the button is drawn highlighted. (false) [R/W]
contents* list of integer The coordinates of the vertices.
```

Return value:

```
boolean The current state of the button.
```

Example:

```
-- A triangular sticky button
{class:poly sticky button, bounds:[195, 30, 340, 150], contents:[0, 0, 145, 60, 60, 120, 0, 0]}
```

A poly sticky button is a transparent polygonal area, similar to a poly push button, that behaves like

Class poly radio button: a radio polygon area

Properties:

```
class class This property must be set to poly radio button.
bounds rectangle
enabled* boolean, integer or list [W]
value* boolean If true then the button is drawn highlighted. (false) [R/W]
contents* list of integer The coordinates of the vertices.
```

Return value:

```
boolean The current state of the button.
```

Example:

```
-- A triangular radio button
{class:poly radio button, bounds:[195, 30, 340, 150], contents:[0, 0, 145, 60, 60, 120, 0, 0]}
```

Saved: Tuesday, February 17, 1998 11:37:29 PM

A poly radio button is a transparent polygonal area, similar to a poly push button, that behaves like

Class movie controller: a QuickTime movie controller

Properties:

```
class class This property must be set to movie controller.
bounds rectangle
enabled* boolean, integer or list [W]
contents alias The file containing the movie.
flags* integer Various flags that control the display of the movie controller. (2)
```

Return value:

```
null This class of item returns nothing.
```

A movie controller dialog item enables the display of any standard QuickTime movie within a dialog. T

The optional flags property is constructed by adding any combination of the following values together

- 1 Add this value to flags to make the movie controller place the movie into the upper-left corner of
- 2 Add this value to flags to make the movie controller resize the movie to fit into the display recta
- 4 Controls whether the movie controller uses a badge. If you add in this flag, the dialog item displ
- 8 Controls whether the controller portion is visible. If you don't add this flag, the dialog item dis
- 16 Specifies whether the dialog item displays a frame around the movie as part of the controller. If y

Class font spec: a text font specification

Properties:

```
name* string Name of the font family for this font spec. (" [maps to the system font]).
size* integer The point size of the text. (0 [means use the system font size which is usually 1
style* style An integer, style name or list of style names indicating the style of the text. (pla
color* RGB colourThe colour of the text in [red, green, blue] format, where 0 is no colour and 6553
```

Example:

```
-- This is the default system font spec:
set font1 to {name:"", size:0, style:plain, color:[0, 0, 0]}
-- It is equivalent to this font spec on Roman script systems:
set font2 to {name:"Chicago", size:12, style:plain, color:[0, 0, 0]}
-- This font spec produces this style of text
set font3 to {name:"Geneva", size:9, style:[bold, underline]}
```

A font spec can be used to override the default settings for the context (and all sub-contexts) within

Note: Currently the color property is not used by control items which always display text in black.

Note: A later release of DD may include a justification property in font spec objects.

The Global Font Spec Table

This section describes the global font spec table (font table) that is specified via the with fonts pa

The font table is specified as a list of font specs and null objects. A null indicates that that font

The tree diagram below shows the font spec inheritance hierarchy in a graphical form. Font spec prope

Example 1:

```
dd install with fonts [null, {name:"Helvetica", style:bold}, null, null, {name:"Geneva", size:9}]
```

This will cause the names of all control items to appear in 12pt Helvetica-Bold and all other text to a

Example 2:

```
dd install with fonts [{name:"Times"}]
```

This will cause all text within the subsequently created DD windows to appear in 12pt Times-Roman, reg

Global Font Spec Hierarchy

Dependencies

This section explains how dependencies (i.e. the enabled property) works. It does not need to be unde

Item dependencies allow interactive dialog items to be dynamically enabled or disabled automatically

There are four types of dependency: none, initial, single item and multiple items. By default a dialo

Negative values indicate inverse dependencies. i.e. if item 4 is a check box then an item with proper

Values with a magnitude greater than 255 are used to indicate a dependency on a sub-item (such as a ra

```
on SubItem(itemNo, subItem)
```

```
return itemNo + subItem * 256
```

```
end SubItem
```

A multiple item (complex) dependency consists of the enabled property being a list composed of an enum

How an item's on/off state is determined depends on the item's class, as follows:

Class	On when	Off when	Comments
push button	never	always	actionless buttons only
push button	after being clicked	before being clicked	sub-dialog buttons only
push button	action returned ≠ null	otherwise	handler script buttons only
check box	checked	not checked	
radio button	highlighted	not highlighted	
radio group	never	always	
radio group button	selected	not selected	
generic control	value ≠ 0	value = 0	
pop up	never	always	

Saved: Tuesday, February 17, 1998 11:37:29 PM

pop up item selected not selected
 list box has a selection has no selection
 list box item selected not selected
 text field contains text empty
 password field contains text empty
 static text never always
 group box never always
 pict never always
 icon never always
 dummy dependencies evaluate to true dependencies evaluate to false
 color picker never always
 gauge never always
 icon button highlighted not highlighted
 poly button highlighted not highlighted
 movie controller never always

Error Codes

Dialog Director may return a number of error codes that are not part of the standard Mac OS/AppleScript

Code	Description
1	The user terminated the dialog by typing <command> + <control> + 'Q'.
2	The script tried to open more dialogs than the maximum allowed.
3	The script called dd interact with user when there were no open dialogs.
4	The global font table is initially too long or has overflowed.
5	QuickTime is not installed. Unable to use movie controller item.
6	Too many, too few, or odd number of polygon coordinates in poly button contents.
7	The Text Services Manager is not installed. Unable to use floating windows.

Dialog Dumper

Dialog Dumper is an applet which is a part of the Dialog Director distribution. Dialog Dumper allows Many thanks to Christopher R. Green for his help in the development and testing of Dialog Dumper.

What You Get

Dialog Dumper is written entirely in AppleScript and, naturally, uses Dialog Director to provide its u

What You Need to Run

To use Dialog Dumper you are required to have the following scripting additions installed on your comp

- Dialog Director v0.7
- Resource Utilities (included with Dialog Director)
- Jon's Commands, from:
<ftp://mirror.apple.com/mirrors/gaea.scriptweb.com/applescript/osaxen/>
 or
<ftp://ftp.cadence.com/pfterry/applescript/osaxen/>
- Programmer's Tool, from:
<ftp://mirror.apple.com//mirrors/gaea.scriptweb.com/applescript/osaxen/pgmTool.sit.hqx>
 or
<ftp://ftp.cadence.com/pfterry/applescript/osaxen/pgmTool.sit.hqx>

What It Does

Dialog Dumper reads each 'DLOG' resource in turn from a resource file and processes its contents and t

This release of Dialog Dumper supports the following Dialog Director features through the user interfa

- choice of Auto Dialog or Live Dialog code generation
- optional greyscale dialog look
- optional timeout for Auto Dialogs
- optionally change the default text font
- optionally change the default text size
- optionally change the default text style.

Dialog Dumper uses the following information from the 'DLOG' resource in its code generation:

- the name of the resource (prefixed with "d") for the name of the variable containing the dialog rec
- or the resource ID prefixed with "dlog" if there is no name;
- the DLOG's boundsRect for the dialog bounds;
- or the boundsRect's width & height for the dialog size if the top-left is (0, 0) or its auto positi
- the DLOG's title for the dialog's name;
- the DLOG's procID for the dialog's style;
- the DLOG's close box option for the dialog's closeable property;
- and the DLOG's DITL ID to retrieve the 'DITL' resource for the dialog's contents.

Dialog Dumper generates code for the following dialog item classes. This information is extracted fro

- push button
- check box
- radio group - by grouping together adjacent radio buttons
- static text
- text field
- icon
- picture
- group box - from user items and control:title Æ name procID = 160 Æ primary group procID = 164
- pop up - from control: title Æ name value ≠ 0 Æ value title ≠ "" and max ≠ 0 Æ name width: max
- list box - from control: procID = 352 value ≠ 0 Æ value
- gauge - from control: value ≠ 0 Æ value max - min Æ max value.

How to Use It

Saved: Tuesday, February 17, 1998 11:37:29 PM

To use Dialog Dumper simply drop the resource file containing the dialog(s) on the Dialog Dumper appli

See elsewhere in this User Guide for details of the meanings of the settings. Make your settings and

If the Live Mode option is selected then the script generated contains a loop that repeats until a pus

Introductory Examples

1. Very simple dialog

```
set dlog to {size:[320, 95], contents:[~
  {class:push button, bounds:[250, 65, 310, 85], name:"OK"}, ~
  {class:push button, bounds:[170, 65, 230, 85], name:"Cancel"}, ~
  {class:static text, bounds:[10, 10, 310, 10 + 32], contents:"A very simple dialog box."}] ~
  }
```

```
set dVals to dd auto dialog dlog
```

This creates a small dialog window in the centre of the main screen containing some static text above

The OK push button is item 1, the Cancel push button is item 2, and the static text is item 3 because t

```
{true, false, null, {416, 397, 736, 492}} -- [ OK button, Cancel button, static text, dialog window
```

Clicking the Cancel button yields the result:

```
{false, true, null, {416, 397, 736, 492}}
```

As you can see, dd auto dialog returns a list of four items, one for each dialog item passed to it via

```
set dVals to dd auto dialog dlog
```

```
if item 1 of dVals then
  -- OK was pressed
else if item 2 of dVals then
  -- Cancel was pressed
end if
```

Alternatively you can write the code like this:

```
set [ok, cancel] to dd auto dialog dlog
```

```
if ok then
  -- OK was pressed
else if cancel then
  -- Cancel was pressed
end if
```

Note that in this example the OK button (being the first push button) defaults to the default button a

2. Simple password dialog

```
property dPassword : {size:[260, 95], contents:[~
  {class:push button, bounds:[190, 65, 250, 85], name:"OK", enabled:3}, ~
  {class:push button, bounds:[110, 65, 170, 85], name:"Cancel"}, ~
  {class:password field, bounds:[10, 36, 250, 36 + 16], name bounds:~
    [10, 10, 250, 26], name:"Enter Password:", value:""]} ~
  , timeout after:60}
```

```
set [ok, cancel, thePassword] to dd auto dialog dPassword
```

```
if ok then
  display dialog thePassword -- OK was pressed
else if cancel then
  beep -- Cancel was pressed
else
  display dialog "The dialog timeout was reached." -- Timeout
end if
```

This creates a small dialog window in the centre of the main screen containing a password field below

After typing some text and selecting it the dialog window might appear as below. Note that any charac

Changing the above script to set the grayscale parameter to true and the initial value of the password

Saved: Tuesday, February 17, 1998 11:37:29 PM

```
{class:password field, bounds:[10, 36, 250, 36 + 16], name bounds:-
  [10, 10, 250, 26], name:"Enter Password:", value:"Test"]}
```

...

get dd auto dialog dPassword with grayscale

should produce a dialog window that initially looks like this (if the Appearance Manager is installed)

Contents of 'Examples' Folder

All the example scripts expect that Dialog Director v0.7 and all the standard scripting additions are

File Description Requires Alert.as Utility that Displays a new (Mac OS 8) style alert.

This section lists any known problems and limitations with this release of Dialog Director (in no particular order):

- There is a maximum limit of 255 dialog items per dialog window.
- A maximum of 7 dialogs may be open at a time (per DD environment).
- Strings used by DD are limited to 255 bytes (characters) except for the text field, password field and text area.
- Many dialog item properties (too many to list) are not available via dd get and dd set. More will be added in future releases.
- When in use DD consumes at least 70K of memory from the application heap in whose layer the dialog(s) are running.
- Some applications do not take kindly to other applications opening windows in their application layer.
- If a live dialog window is left open after the script returns control to the calling application (script ends), the dialog will remain open.

dd install -- plus any options

try

```
<code to manage my Dialog Director windows>
```

```
dd uninstall
```

on error

```
dd uninstall
```

end try

- If dd interact with user... is not called frequently enough while a live dialog is active, mouse clicks will be ignored.
- If an application uses DD to create floating windows and then quits without calling dd uninstall the dialog(s) will remain open.
- Due to the manner in which DD v0.6 handles backward compatibility with DD v0.5.1 properties that have been added in v0.6 will not work in v0.5.1.

Other Scripting Additions

Scripting additions included as part of the Dialog Director v0.7 package:

- Resource Utilities Read, write, get & set info for any resources and read write any AppleScript dictionary

Scripting additions that are not currently part of this package:

- Picture Utilities Read, write, print, join, resize and get info about Quickdraw pictures.
- Display Picture Display a Quickdraw picture in a modal, scrollable window with magnification.
- Make Picture Create a picture from within AppleScript. Supports rectangles, ovals, lines, text, etc.
- String Width Calculate the width, in pixels, of a text string of given font, size and style.
- Set Info For Set info for a file or folder (counterpart of info for command).
- Font Info Get ascent, descent, leading and height of a given font, size & style.
- Print Utilities Create print jobs and print multiple pages containing text & pictures.

```
Ç htB' #A#H&K ~~~~F f ~
```

~~~~~  $fE + \hat{a}$  ~~~~~  $\sim \sim \mathbf{1} \mathbf{E} f \rightarrow$   $\mathbf{D} \rightarrow$

~~~~

~~ÃÃ ~~~ ~~~ff ~~~33 ~~~ ~~~ff~~ ~~~ffÃÃ ~~~ff~~ ~~~~~ ~~~ff33

~ ~ ~ ~ ~ ~ ~ ~ 5 ~ ~ ~ ~ 5 " (E ° θ ~

~ ~ ~ È Z " ŮÄŮÄ ,

Helvetica

. + Dialog ~~~33

~ ~ ã"â"1 ŷ\$ 3"ð D°

~ ~ ~ ~ ~ ~ ~ ~ ~~see~~

~ , ~ Ë Z " ŮÄŮÄ Button+ó Pŭŝh

~ ' ~ © Z " ŮÄŮÄ (Grouped 103

Yw ~ ~ ~ ~ à " à 1 i x 3 " ð ° ~ ~ ~ ~ ~ ~ ~ ~ ~ ~

~ , ~ 1 Z " ŮÄŮÄ (
GenericControl†ó ~~~~33

~ ħ ~ j Z " ŮÄŮÄ) Box16iŠč33

~ ħ ~ U Z " ŮÄŮÄ (FieldText33

~ ' ~ U Z " ŮÄŮÄ (Páññwóřč~33

~ 1 ~ + Z " U Ä U Ä) Box†ó ~ ~ ~ 33

~ , ~ Z "ÛÄÛÄ) Pict+ó ~ ~ ~ ~ 33

~ , ~ z "ÛÄÛÄ(\$ Icontó ~ ~ ~ ~ 33

~ 1 ~ @ Z " ŮÄŮÄ (Tstáčřč3

~ 1 ~ z " ŮÄŮÄ) Graphic+ó ~ ~ ~ ~ 33

mī ~ ~ ~ ~ à " 5 " 1 [" 2 = 0 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~


~~~~ z "ÛÄÛÄ ) Dialogtó ~~~~33

~ ~ ã"ã" 1 «B" = ∂

~ ~ ~ ~ ~ ~ ~ ~ ~~see~~

~ 1 . i z " û Ä û Ä ( Dummy + ó

[illegible]

~ ~ ~ Z " U Font Spec (ó ~ ~ ~ 33

~ ă " ă " 1 P 1 5 « " ∂ A °

~ ~ ~ ~ ~ ~ ~ ~ ~~see~~

^ì,≠      Z      "      ŮÄŮÄ (

MovieController†ó



Saved: Tuesday, February 17, 1998 11:37:29 PM

---

m S ~ ~ ~ ~ ∂ 5 " M ° " Y l ~ à " ä " I 3 8 1 t « ∂ # °

~ 1 . 0      Z    " ŮÄŮÄ    Button+ácon

Saved: Tuesday, February 17, 1998 11:37:29 PM

---

m h ~ ~ ~ ~ 0 5 " D ° 0 " n l f y ~ 3 4

˘ ˙ ˘ æ      z      " ŮÄŮÄ      (

Radio Button†6

ŸB ~ ~ ~ ~ ∂ 5 " G ° " á l ~ ä " ä " B 3 l ç « # ∂

~ 1 ~                    z   " üÄüÄ )           Pop-Up + ó

mÅ      ~ ~ ~ ~      ∂ 5 "      = °      ∂      "      ~~Y~~ eDà "      "



˘ ı ˘ Ë      z   " ŮÄŮÄ      )      Control+ó

Saved: Tuesday, February 17, 1998 11:37:29 PM

m ~ ~ ~ ~ ∂ 5 " < • úÿÓ ä"ä"BBÿç 3 ð#°

Saved: Tuesday, February 17, 1998 11:37:29 PM

Icon List Box



~i,◊      z      " üäüä )

Color Picker†6

m ) ~ ~ ~ ~ ∂ 5 " F ° " / 1 ~ à " ã " İ 3 8 1 J « ∂ # °

~ 1 , - z " ûÄûÄ ) Guage t ó



Saved: Tuesday, February 17, 1998 11:37:29 PM

---

m > ~ ~ ~ ~ 0 5 " : 0 0 " D I ~ 0 H

~ ' . ò      Z    " ŮÄŮÄ ( Push Button    London ~ ~ ~ 33



---

~ ' . É Z " ŮÄŮÄ ( Toggle BitContent ~~~33



---

~ ' , Y Z " ŮÄŮÄ ) Radio Button + Ó

Ÿ ß ~ ~ ~ ~ ∂ 5 " Q ° " ò Ÿ ~ à ¨ ä ¨ ı 3 ß Ÿ û ß #

~ ' , n z " ŮÄŮÄ ) Sticky Button +ó



ÿ 1 ~ ~ ~ ~ ∂ 5 " Q ° ∂ " n Ÿ Ö œ ? à #

---

~ 1. D Z " ÜÄÜÄ )  
Poly Buttonó

Saved: Tuesday, February 17, 1998 11:37:29 PM

m o ~ ~ ~ ~ 0 5 " D ° 0 " " — . ~~SL~~ ~ # # #

~ ' . D Z " ŮÄŮÄ Push Button Pó 1ŸŸ33



---

~ ' \_ / Z " ŮÄŮÄ ( Toggle Batch ~~~33



---

~ ' . Z " ŮÄŮÄ Radio Breeze



ÿ ° ~ ~ ~ ~ ∂ 5 " Q ° " ï Ÿ ~ à ¨ ä ¨ ı 3 ß Ÿ Ů ß #

~ ' . Z " ŮÄŮÄ , stickPottoró

Ÿ Ê ~ ~ ~ ~ ∂ 5 " R ° ∂ " — Ÿ ⊕ — #

Saved: Tuesday, February 17, 1998 11:37:29 PM

~~~~~ c+ã ~ ~ ~ ~~~~~ 1@ ~ @

Helvetica

- (M + Large + Icons @ ~ " Ö f K Ñ

323 2 @ ~ " i f â 5 11 2+ (

Saved: Tuesday, February 17, 1998 11:37:29 PM

| | | | | | | |
|----------------|---------|-------|----------|----------|--------|-----|
| 1616 | @ ~ " √ | ~ f √ | Mini | + | x 12 | @ ~ |
| Black & White, | Monaco | +Z | ICN#) < | ics#) < | icn#rs | |

Helvetica

- (M + Large + Styles N ~ " Ö f K Ñ

323 2 N ~ " i f â 511 2+ (

Saved: Tuesday, February 17, 1998 11:37:29 PM

| | | | | | | | | |
|---------------|-----|-------|-------|------|-----|---|-----|---|
| 1616 | N | √ | √ | Mini | + | x | 122 | N |
| 0 Pixels Deep |) c | 0) < | 1) < | 2 # | (/ | | | |

Saved: Tuesday, February 17, 1998 11:37:29 PM

1 Pixel Deep) c 4) < 5) < 6 # (=
2 Pixels Deep) c 8) < 9) : 10 # (K
3 Pixels Deep) a 12) < 13) < 14 " ŮÄŮÄ† Ç ~ " 5 < Ů

~ ~ ~ ~ ~ ~ ~ ~ / 4 0 n 2

Helvetica

-



SystemFont)Z

Font Spec #1)Z

Saved: Tuesday, February 17, 1998 11:37:29 PM

```
Font Spec #2)Z pushbutton - name      $ ỳ Geneva
(  Z global)Z
control names
+Z  check-name    $6 ỳ* button-name  6H ỳ* graphic content
```

Saved: Tuesday, February 17, 1998 11:37:29 PM

Font Spec #3)Z statictext - contents 1~ ȳ

(vȳ

static labels

+Z group name ~ê ȳ* up -pmp

~)E (always Font Spec #3) êç ȳ

(û text name



†ç)L (always Font Spec #3) çȳ ȳ

(Ø password- name

K†)g (always Font Spec #3) Δ ȳ

(i ȳ

Saved: Tuesday, February 17, 1998 11:37:29 PM

Font Spec #4) Z popup - value Δ  
(-¥ data values
+Z text - value ýí ý* passwordvalue Í, ý* bokiste
* usefontstó ~~~~~ 4 /ø " - , ° • • ø # 0

~~~~ôô Ê 7 ~ ~ ~ ôô 7 ÊôÅNôô 7 ÊHH k ~ µ1 ~ ~

~~~~  
~~~ÃÃ ~~~ôôôô ~~~ôôff ~~~ôô33 ~~~ôô ~~~ff~~ ~~~ffÃÃ ~~~ffôô ~~~ffff ~~~ff33  
~ÃÃ[] + ~ ~Ã~+~ ~Ã~+~ ~~~~+îú+î~+ V~~ ~~~+î·ú+î·.+ V~~\$ ~~~+î,









~~~~~ ñq+â ~~~~~ ~TǒqPñ o p "ûÄûÄòÅR

~Å·Å·≥ - Å~Å~±~ #†Ç ~~~~qð ñ#º ~~~~~~"P g @#º

„ Ç ~ ^ Q ! " ŮÄŮÄ ! ,

Helvetica

. * 3Item! * 2Item! * Item ~) E: b r

~~~~~

qta

~~~~~

10


~~~~~È ~, ~- ~\*~Ä~\ ~\*~Ä~é~ ~.~ ~~~~~~Ö.



~ÅÄÅÄÖÄ Å~Å~Î~ #+Ç~H¿ - r İ r / ~ ~ ~

~~~~~ q+å ~~~~~ 1ö~~q o "ûÄûÄðÄ


~ Å Ä Å Ö Ä Å ~ Å ~ Î ~ # + Ç ~ S O ñ O Ü ~ ~ ~ ~

~~~~~ n t a ~~~~~ I ~ n

~~~~

~~ÃÃ ~~~ ~~~ ~~~33 ~ ~~~ ~~~ÃÃ ~~~ ~~~~~ ~~~33


~~~~~î^ -.~-.~^ v°~ v~ê^ ~^~^ ~.~^ ^ ~~~~~~.~ ^ -.~





f p` p @) i` ><~@JÆ Σ KÊ t Á\*

Ⓒ ∞L\$∞\$H\$œ\$†% % %6%;%m%√%Σ%π%<%Ã%◇& & & & ) & +




Saved: Tuesday, February 17, 1998 11:37:29 PM

\* \* < \* E \* R \* W \* k \* s \* Ä \* Ü \* ö \* o + + + + " + O + ; + F + P + U + C + x  
/ 2 / 8 / E / J / å / ï / ü / # / π / Æ / ' / Ī / ¯ O O N O \ O ï O ™ O Ê O Ĩ 1 \$ 1 \* 1 ` 1 i 1  
: : : : : : # ; 3 ; 7 - Ů - - Ů - Ů - Ů - - Ů - - Ů - - Ů - - Ů - -

S S - - ū - ū ū ū - - - - ū ū ū ū Ō Ë , , , , - - - - ū - - - -



---

[ [ [ [  [ [ [ \$ [ & [ M [ R ' 1 ' 1 ' 1 ' 1 ' 1 ' 1   $\partial$ Y  Y\$Yt

---

\_\_+\_4\_£\_Ø\_fl\_ËaVaaa´a∂aΣa¬a√aÄa,aÎaÓa~b\_bnbpb~bÄbåb

e e ã e Ÿ æ f † f Ó g g ) g . g : g r g | g É g ã h D h Q h à h h " h e h , i . ~ ú





P i i 1 i 4 i 5 i : i ; i = i > i T i U i X i Z i \ i ^ i \_ i É i â i Ó j j ; j I



o o o o o o o ] o ^ o à o ò r r r r r r r { r Ä r å r ð i n Σ r æ r



---

Ä Ä Tsgsä sã s é s è s ê s ë s ì s î s ö s õ s t s

---

t t t t&t)t\*t/tOt2t3t9t:t?t@tBtCtHtItMtNtPtQtUt



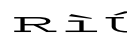


\u u u u u"u#u%u&u,u-u3u4u6u7u;uku1v]vcv¶vÆv/v‡v



w | | | | | æ } | § |  
} } # } % } . } 1 } 9 } Ñ } ê } + } ™ } Ó } · ~ ~ ) ~ R ~ ` ~ b ~ w ~ y ~ ~ Å ~ á ~ â ~







W O

O

⌘ ⌘ ⌘ ⌘ ⌘ ⌘( ⌘ .⌘2⌘4⌘: ⌘ ⌘A⌘⌘D⌘G⌘H⌘S⌘T⌘V⌘V





Ä Ä ^ö lönötövöÑöÖöáöàöîöîön



Ä

Ä

Uü7ü=üKü+ü™üØü<sup>a</sup>ü ü~

+ + " + # + u + + Ū + ~ ° ° ° A ° E ¢ 8 ¢ D ¢ [ ¢ k £ ¶ £ ß § ¤ ± • ¥ • fl •



-

---

Ä

Ä

L ' È ' Ì ' Ö ' ° ' ¬ ' " ' / ' -- ' Σ--

~~Æ~~  
~~Æ~~<sup>∞</sup><sub>μ</sub><sup>∞</sup>æ±5±C± ] ±r±ı±†±È±ı≤ | ≤ā≤ı≤s≤ ≤a≥/≥>≥K≥Y≥e≥l≥r≥







$\sqrt{\sim} \approx i \approx j \approx r \approx s \approx \{ \approx \infty \approx \ddot{A} \approx$







Ä Ä\Ã6Ã7Ã;Ã=Ã>ÃCÃHÃMÃ\ÃgÃä.ÃæÃæÃæ,ÃË:Ö  
æ æ æ#æ-æ1æ8æ@æÇæâæëæõæßææææ...æ-æÿ-æ%-æ/-æm-æy-æ-

" ! " ' ~ · ~ ŮİŮÁ~Ů , , Ů , Ů , Ů , Ů , Ů , Ů , , İ~Ů , , Ů , Ů , Ů , Ů , Ů , Ů , , İİ·İÁİİ~ŮİŮ

÷ ÷ Õ ÷ Æ, 7 ? Ç -- ° ± Í Ÿ Ÿ Ÿ Ÿ ' Ÿ ( Ÿ \* Ÿ - Ÿ 0 Ÿ 4 Ÿ 7 Ÿ ; Ÿ @ Ÿ D





È:  
ÈUÈXÈYÈeÈfÈhÈjÈnÈpÈsÈuÈwÈzÈÇÈàÈçÈéÈôÈõÈ°È£È•ÈßÈ©È  
È È È È È#È(È)È3È5È;È=È?ÈAÈCÈEÈGÈIÈKÈNÈRÈTÈY

ô ô ô\_ôïôçôüôéôûôæôø · ~ · ùû · ~ ôÈô„ · > „∅·∅· · ·—À· · ·—À—·—À·Àû

~ ~ ~ ~ ~ ~ ~ \$ ~ % ~ & ~ \* ~ 8 ~ 9 ~ F ~ G ~ H ~ T ~ Z ~ μ ~ ~ B ~ :

T ^ æ · · ~ ~ ~ ~ · ~ · ~ · Û Ò · · · · î · Â Â % · % · % · fl fl · fl · fl · fl · fl ·





==



I-I



$$(\mathbf{N}^{\mathrm{p}} \mathbf{F}^{\mathrm{p}})^{-1} @$$
$$\geq \infty$$
 $\mathbb{I}\mathbb{P}$  $a_i$ 

140e



U



Saved: Tuesday, February 17, 1998 11:37:29 PM

---

|   |   |    |    |    |    |    |    |   |    |    |    |    |    |    |    |
|---|---|----|----|----|----|----|----|---|----|----|----|----|----|----|----|
| 2 | 2 | 2  | 2  | 2  | 2  | 2  | 2  | 2 | 2  | 2  | 2# | 2% | 2+ | 2, | 2. |
| 6 | 6 | 6! | 6D | 6N | 6V | 6ò | 6¢ | 7 | 7" | 7# | 73 | 75 | 7; | 7= | 7  |



---

@ @ö @ä @ã @õ @ù @f @x @Σ @-√ @« @... @E @œ



---

|   |   |   |   |   |   |    |    |    |    |    |    |    |   |
|---|---|---|---|---|---|----|----|----|----|----|----|----|---|
| M | M | M | M | M | M | M# | M% | M& | M( | M) | M+ | M. | M |
|---|---|---|---|---|---|----|----|----|----|----|----|----|---|

---

Q Q Q Q5 QN QZ Qa Qg QÉ QÑ Qñ Qò Q† Q´ Qµ

Saved: Tuesday, February 17, 1998 11:37:29 PM

---

|   |   |   |               |    |               |    |                |    |    |    |    |    |    |    |
|---|---|---|---------------|----|---------------|----|----------------|----|----|----|----|----|----|----|
| S | S | S | S             | S  | S             | S  | S              | S  | S  | S  | S* | S` | Sp | S  |
| Z |   | Z | <del>ZZ</del> | Z# | <del>ZZ</del> | Z( | <del>Z</del> * | Z, | Z. | Z0 | Z2 | Z5 | Z7 | Z8 |

---

P

P

Ä

}  
} } # } \$ } ' } ( } / } o - ùóóùùë:óë:ó

---

|   |   |     |                |     |    |    |    |    |    |    |    |    |    |   |
|---|---|-----|----------------|-----|----|----|----|----|----|----|----|----|----|---|
| ~ | ~ | ~2  | ~=             | ~F> | ~F | ~J | ~U | ~  | T  | `  | ù  | ß  | ª  |   |
| Ö |   | ÖÖ% | Ö <del>Ö</del> | Ö1  | Ö3 | Ö9 | Ö; | Ö> | Ö@ | ÖB | ÖD | ÖG | ÖI | Ö |

---

@

!

W

ö â ≥

ö â ñ Ω ö

õ  
õ      õõ   õ"   õv   õÅ   õñ   õç   õ• | õ+   õ<   õõ   õõ °   ú4   ú8   ú



---

X P H @ 8



U u . u1 u3 u6 u8≤ u6 uū>uCuūN uD







Ä

Ä

Ä8

@8

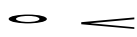
¢ ¢ ¢ ¢ ¢ \$ ¢ % ¢ O ¢ 2 ¢ 8 ¢ : ¢ = ¢ ? ¢ A ¢ C ¢ F ¢















P

ED

[illegible]



h

T

~

(

<

< <

< <

( <



— ( <

\_\_\_\_\_





< <

< <

< <

< <

⌂ ( ⌂ ) ~ ( < M ( <

~      ( <      ~      ( <

< <

h

T

< <



U

^

(

@

(

<

Û ^ ( @B\$ \$ „ % %Q%~%fl&S& &Û ' ' D ' j ' ï ' Ÿ ( 4\*(\*3  
C <



~ ~ ~ ( <





Saved: Tuesday, February 17, 1998 11:37:29 PM

X= =q=¿>><>X>ß>...?a?Ã@Ç@«Að2B BNB™CcCfC~D&DÉfB F  
SDSiSwSπS~T8T@TKTöt^T„TÚVLVpV@V.WEX XYXç.....^ò·İ·İ.....İ.....İİÁ^,>ÿ

< <



< <

< <

< <

-

< <

< <





< <

~ ( <





FXçY YtY...Z!Z\*ZhZ¶ZË[@[R[S[}[~^.\_"aRdLfıgÉhrhΩh"1

ä m ä ¨ ä µ ä œ ä ÷ ä    Œ   |   ~    Ů Å   Ç È Ö Ů ä ~ Å Ä ä  
ä - ä ; ä \_ ä r ä i ä ß ä # ä    ä " ä > ä , ä ~ è R ê    é O ê ç ê % ê Ì ë    ë l ë m î î 3 î m

+ + "+#°Açnç}·^^^·^^^·Öïïïfl·····^^·Öïïfl···^Öïïfl·····^^·Öïïfl···



o

h

T

≠

O h T .

È:  
ÈUÈÜÈ√È ! ÈÖÈ∞È±Î≈Ï . ÓıÓπÖ≈Ö˘ Ô



ô\_ôüô、 ¼ ± ∅ò ò.·····ó·óèè,,èèèèfiŸó·····Π.....ó ° ° ° ° óèè



) )J )q ) )œ )ÿ \*- \*\ \*è \*ê ,Y ,É Π è i, «-Ó /. ,

@O @X @Ñ @" @" B9 BÇ Bé BÃ Bfl C C^≤ CÐ FC< FUI DÀ

Saved: Tuesday, February 17, 1998 11:37:29 PM

---

V+ Vã Vå XE Xs X X° Xœ Æ ^Y Ÿ CYM ZŸ Ç ZŸ? \ó \Ã \ÿ

Ω ^ \_N ` ¢ bN c9 eē eÛ fO f¥ g g" gµ gfl h h\

l' n' q> r. r~ s sm ti tj tu tũ u|ë |ëi} u\$ }@Hl w>

° ¯D Ä@l ,† ' ¸ °∞ äÄÄ h<sup>≥</sup>  
°b.





UπΓπ π π\$πeπfπgπpπRπΘπ™π≤ ∫ ∫ < ∫ = ∫ H ∫ { ∫ | ∫ } ∫ ζ a { a | a }



ÙT à Ç# Æ Æd Æ© ÆT Æfl Õ" œ[Ω — Æ — à' — ê, — í — o

Û! Å Example Normal (No cap)

Property Def Details Detail(Indent) ReleaseBullet Details(Indent-2) Normal(Indent) Event Def )

( ( + 8  
8 ~ O ~ -  
P - -

+ 8  
- ~ L

h T — x' @x @Q@FÀMfl ' @  
p ð ̄ \$ ^ ) C . & 1 • 4 F' = Ũ C Û I ö P U = Y ~ a l à  
¬&o&p&q&r\*X\*ä\*ë\*ù\*Ö\*Ø2c2d2e2f2g2h9F:E:F:G:H:~:£?~  
~ , ~ " d , 6 G { ‡ HH  
ê A » .